31 Mar 2021 — More on NP Completeness

PRELIM 1 QUINTILES

	Thurs	Fri
80%	46.5	45.6
60%	45.0	43.0
40%	42.6	40.0
20%	36.0	37.4

Grades to be released on CMS after today's lecture.

Designing a reduction from 3SAT to INDEP SET.

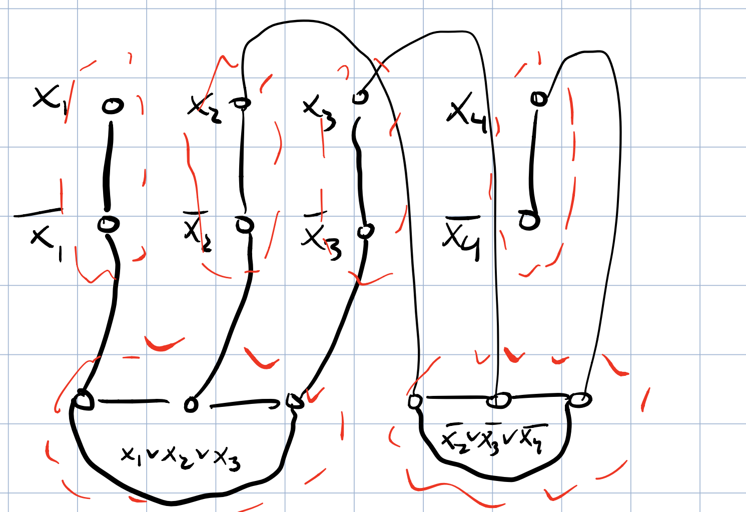
That means: given a 3SAT instance (variables and clauses) we construct a graph G and a number k such that G has a k-element ind set $\iff$ the given 3SAT instance is satisfiable.

Step 1. For each literal (x_i or $\overline{x_i}$) create a vertex. For each clause create 3 vertices

Step 2. Join x_i to $\overline{x_i}$ with edge. Join the ~~3~~ vertices of each "clause gadget" into a ~~3-cycle~~. clique. Attach the vertices of this ~~3-cycle~~ clique to the negations of the literals in that clause.

Step 3. Set $k = (\#\text{ vars}) + (\#\text{ clauses})$

$(x_1 \vee x_2 \vee x_3)$
$\wedge\ (\overline{x_2} \vee \overline{x_3} \vee \overline{x_4})$



$k = 4 + 2 = 6$

Running time of reduction: if 3SAT has n vars, m clauses

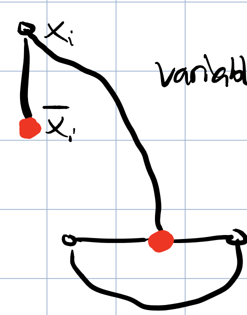
Step 1: $O(n+m)$
Step 2: $O(n+m)$ } $O(n+m)$
Step 3: $O(1)$

Correctness: G has a k-element ind set $\iff$ the given 3SAT instance is satisfiable.

($\Leftarrow$) Show that given a satisfying truth assignment we can get an indep set of k elements.

GADGETS WORK AS INTENDED

The graph is composed of $k = n+m$ "gadgets"



A satisfying truth assignment gives us a way to choose one vtx from each gadget.

E.g. if $x_i = F$ and clause illustrated above contains $\overline{x_i}$.

No two elements of this set of k vertices have an edge between them

($\Longrightarrow$) if The graph has an indep set of k vertices, they must all belong to different gadgets. (every 2 vertices in same gadget are connected by an edge)

$\Longrightarrow$ exactly one vertex in each gadget

$\Longrightarrow$ let a truth assignment of $x_1, \ldots, x_n$ be defined by setting $x_i = T$ if node x_i is in indep set, $x_i = F$ if node $\overline{x_i}$ in indep set.

This truth assignment satisfies every clause b/c the node of the clause gadget that belongs to the ind set identifies a satisfied literal in that gadget.

NO "UNINTEDED" WAY OF USING THE GADGETS

If A and B are decision problems — computational problems with a Boolean answer (encoded as 1 for TRUE, $\emptyset$ for FALSE) then a poly-time reduction from A to B is an algorithm R_{AB} running in poly time such that

instance of A — instance of B resulting from running reduction.

$$\forall x \quad A(x) = B(R_{AB}(x))$$

If such alg exists we write $A \leq_p B$.

Note: This is a transitive relation.

If R_{AB} reduces A to B and R_{BC} reduces B to C then

$$\forall x \quad A(x) = B(R_{AB}(x))$$

$$= C(R_{BC}(R_{AB}(x)))$$

So $R_{AC} = R_{BC} \circ R_{AB}$ is a poly-time alg reducing A to C.

<u>Interpretation</u>: $\leq_p$ ranks problems by computational difficulty.

<u>Def.</u> P is defined as decision problems that can be solved in poly time.

NP is defined as decision problem A s.t. $A \leq_p 3SAT$.

A problem B is NP-Complete if $B \leq_p 3SAT$ and $3SAT \leq_p B$

Note that if $3SAT \leq_p B$ then <u>every</u> problem in NP reduces to B.

$A \in NP$ means $A \leq_p 3SAT$

then if $3SAT \leq_p B$

we conclude $A \leq_p B$.

To show a problem is NP-Complete we must:

1. show it belongs to NP (e.g. by reducing to 3SAT)
2. reduce any NP-Complete problem to it.

Conjecture: $P \neq NP$